Customizability of Visualizations in Web-based learning systems

Abdulmotaleb El Saddik [1], Javier Diaz [2], and Ralf Steinmetz [1,3]

[1]Industrial Process and System Communications,
Dept. of Electrical Eng. & Information Technology,
Darmstadt University of Technology,
Darmstadt, Germany

[2]Laboratorio de Investigación en Nuevas Tecnologías Informáticas,
Universidad Nacional de La Plata,
ciudad de La Plata,
provincia de Buenos Aires
Argentina

[3]GMD IPSI,
German National Research Center for Information Technology,
Darmstadt, Germany

Abstract

In this article, we discuss the usability and customizability issues of interactive animations and simulations in web-based learning systems. The approach presented here extended our component-based architecture to build up interactive multimedia visualization units by the use of metadata.

1. Introduction

Developing animations of computer algorithms & networking protocols is becoming a well-known process for teaching computer science and networking. As the computer systems grow and the interactions became more complex the need to introduce the concepts in a simpler way requires friendly, better and more customizable tools.
The development of a set of animations in Java to show how several data structure algorithms and networking protocols work in different set ups, has accelerated the students' process to fully understand new concepts [Steven Hansen]. Nonetheless the professor is overloaded in his duties since he has to know exactly how to interact with each applet and how to configure them in order to suit best his needs in the lecture. Although such systems can be used in the curricula, they are to some extent unusable. This is due to the number of parameters the professor can adjust. For this reason we think that a useful system should provide professors with an easy to use customization tools. became apparent.
Though the usability factor of these animations is affected by the HCI considerations [2], the main drawback of the tools from a pedagogical point of view was the need of the professors to take into account many details to show the students different scenarios.
In the following we will describe how we solved the

1) usability of the applets (standard layout of the interface, standard menu, unified way to provide help and advice, notes on the demos enclosed with the applet)
2) customization of the applet regarding the professors intention (among other: handling different scenarios, interaction level, explanation level)

The presented paper shows how the aforementioned usability and customization process of the applets suits the need for both of our systems: a) the Virtual Java-simulation Lab (VJ-Lab) [3] for Computer Science Students developed at University of La Plata and b)

the Interactive Teaching BeanKit (itBeanKit) [4] developed at Darmstadt university of Technology.
The rest of the paper is structured as follows: In Section 2 we briefly explain both our system, the VJ-Lab and the iTBeanKit related work and define multimedia Learning Objects (LOs) as well as learning objects metadata. In Section 3, we present an overview of interactive multimedia content and their characteristics, before we introduce dynamic metadata in section 4. Section 5 describes our implementation, and Section 6 concludes the paper and gives an outlook.

2. Interactive Visualization Systems

VJ-Lab

The VJ-Lab [3] is a web-based system that incorporates new web technologies, such as Java in order to develop interactive visualization units for teaching. These units are applets, which provide dynamism to the HTML documents and facilitate the updating of new system versions that is possible because they will be dynamically loaded from the HTTP server.
VJ-Lab allows the learning of some computer science curricular contents by doing. There are two approaches in this learning process:

a) applying high order abilities by using contents and processes simulations. They are more difficult to understand from traditional teaching and learning tools, such as textbooks
b) putting into practice the Java concepts taught and applying theoretical curricular contents by building Java-simulations

The VJ-Lab is conceived to promote the undergraduate students of Computer Science:

- Accessing to the newest technologies in an easy and fast way.
- Training the students in Java technology from a meaningful viewpoint.
- Learning contents with high degree of abstraction by using computer based simulations. These support students with strong difficulties to understand topics like data networking applying the concepts in a real situation.
- Facilitating and automating the complexity of some contents and processes. For example, VJ-Lab allows to simulate the Routing protocols by a single PC.
- Handling a great volume of complexity data by means of simulations (e.g. Hierarchical Tree AVL, Topological Sort, etc.)

During the last years students of the 4th year of the computer Science Department have developed two groups of simulations applets, which are used in the lecture and world wide available:

- Data Network protocols, available at: http://www.linti.unlp.edu.ar/catedras/Laboratorio/1999/applets.htm
- Data Structure algorithms, available at: http://www.linti.unlp.edu.ar/catedras/Laboratorio/1998/applets.htm

iTBeanKit

As software engineering evolves, new elements of this engineering domain emerge. Where raw, undifferenciated, white-box code once was, dynamically pluggable black-box components begin to appear as a result of the enormous growing of the use of Java as a programming language and especially of its accompanied JavaBeans Technology. Our framework, which we refer to as the interactive Teaching Bean KIT (iTBeankit), is based on the black-box component software technology. A detailed explanation of its architecture may be found in [4]. In our framework, an algorithm visualization is more than a mere animation. It describes an environment that elicits active student participation using a careful presentation of information in various media (such as animations, text, static diagrams, audio, video, etc.) with appropriate possibilities to interact with the system whenever a student wants to do it.

The implementation of several multimedia concepts and communication protocols can be found on the web:
http://www.kom.e-technik.tu-darmstadt.de/projects/iteach/itbeankit/html/

3. Offering support through interaction

As we mentioned in the introduction the requirements we are dealing with in both our systems are *interaction* with the visualization and *support* of both the learner and teacher.

Interaction implies that the learner is guided in the sense that he can get feedback if problems emerge. Assuming that the handling of the visualization itself is intuitive such problems can only result from the difficulty of the topics to be learned. The *difficulty* of an algorithm to be animated can result either from the knowledge of the learner which might not be sufficient to understand the topic or from the amount of information presented by the animation. If the user's knowledge is not sufficient to understand parts of an algorithm we offer two possibilities to create the corresponding knowledge: a user can read a short explanation of the part of the algorithm he currently executes or he can invoke the chapter of the textbook explaining the underlying theory in depth. The latter includes search functions to get a more specific way of explanation.

The processing of an insufficient knowledge of a learner is performed in a traditional way by using hyperlinked multimedia documents. The second problem however, the density of the presented information has to be dealt with by another approach, the use of *levels of complexity*. The idea behind a level of complexity is that a user can reduce the information density of a part of an algorithm by splitting the part of an animation he/she is currently using into a particular number of steps which can be understood easier equivalent to a smaller information density. This process is shown in Figure 1. While *C* stands for complexity, the upper index denotes the level, the lower the number of a component.

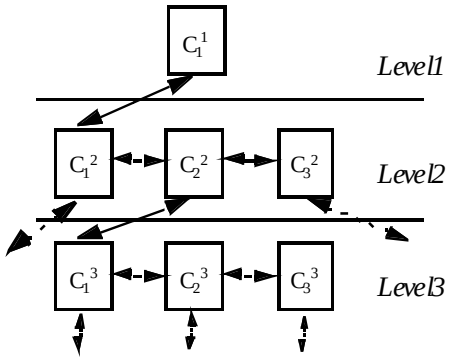


Figure1: Levels of complexity

The first step towards usability of visualization learning materials is to have a uniform graphical user interface (GUI) [2] [4]. The visualization window is divided into three areas: an animation pane, an explanation area, and a parameter pane. The animation pane displays the resulting animation envisioned by the educator. The explanation area displays some hints and information concerning the visualized algorithm. The parameter pane is divided into two parts: interactive utilities pane and VCR. The VCR is algorithm independent, existing in all animations, and allows the end-user to control the progress of the animation. On the other hand, the interactive utilities pane is algorithm and topic dependent, defined by the programmer and customized by the educator. The animation may request intermediate input from the end-user, allowing them to control the path of the algorithm.

In the following we will explain our distinction between the two kinds of interactions the user is provided with: content dependent interactions and content independent interactions.

Content-Dependent Interaction Model

These interactions are strongly bounded with the topic to be visualized:

- Variation of parameters of a running visualization (Applet)
  - Visualization (level of complexity)
  - Animation (speed, background color, foreground color,..)
  - Simulation (interactions by the user interface)

Content Independent Interaction Model

This model represents those interactions, almost all developed applications will have in common:

- Guiding the user
  - Help function
  - Step function
- Explanation Language (at the moment we are supporting: English, German, Spanish)
- Explanation (Errors, Hints, Audio)
- Look And Feel (Java, Windows, Motif)

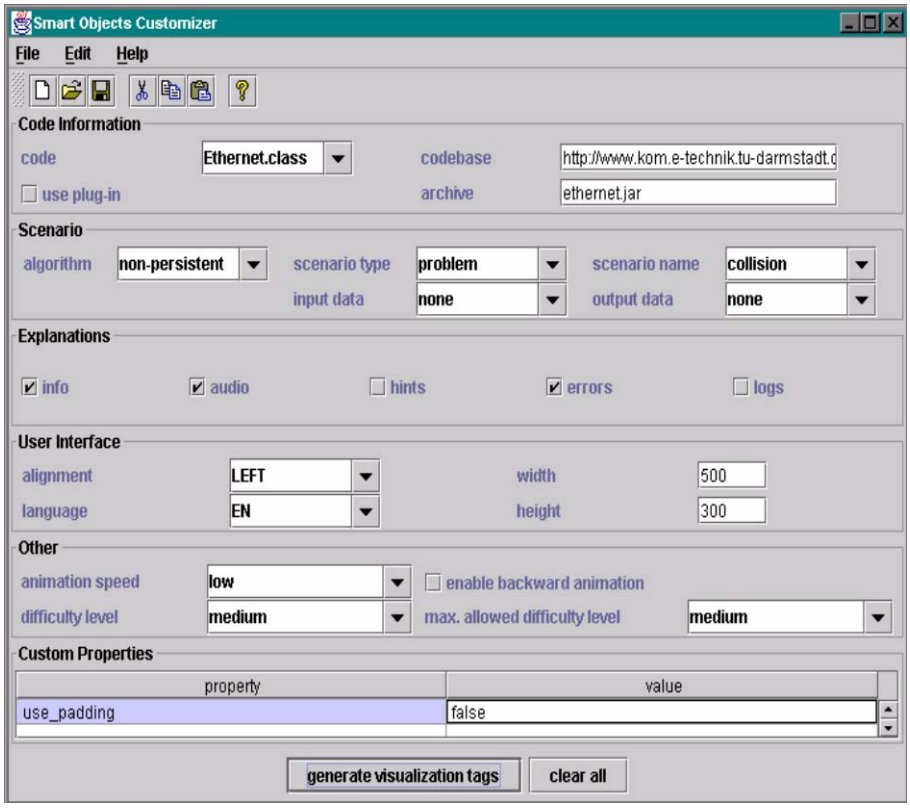
4. Offering support through Customization

A number of directions accompany the software development process, and try to define the way in which software evolves. One way in which software evolves is to push a configuration out onto the user and to make the program as interactive as possible. An alternative way is to defer such decisions until runtime, or to push configuration decisions out into the data. Data themselves become more universal and reusable when they are accompanied by metadata. By metadata we mean data that describe other data, rather than aspects of the application domain itself. Metadata let a person or other programs make sense of the data they describe. These data may become even more general and independent when they are integrated in the code which may travel over the internet. Naturally, these metadata or code descriptions should be objects too. Hence metadata have metadata as well. Instead of using metadata in its general term and traditional role, to create universal descriptions of objects, as it is the matter in digital library systems for example, they are used instead to describe properties. According to a suitable programming language or software system properties might be known as attributes, dynamic attributes, variables, dynamic variables or even dynamic slots. Therefore, runtime mechanisms for accessing, altering, adding and removing metadata or properties at runtime should be provided by the programmer.

We define the term "dynamic metadata" as the description used to adapt the content of an object, and/or to change the behavior of a learning object. As an example of dynamic metadata, we will in the following examine the simulation of the CSMA/CD protocol (Ethernet). To be able to explain Ethernet properly, specific problems have to be addressed, for example the collision of packets on the bus, or the shortframe problem. The key idea behind dynamic metadata [1] is that the same visualization can be used to explain different problems, if it is configured by parameters. In the following we will explain the data structures for dynamic metadata in detail, but to motivate the problem, we provide an example here. A part of the data structure could be a field “PROBLEM”, addressing a specific parameter configuration of a visualization. Concerning the visualization of Ethernet, changing the value of the metadata field "PROBLEM" (being represented in the program as a property) from "Collision" to "Shortframe" may change the whole behavior of the algorithm to be visualized.

As discussed in [1], dynamic metadata contain properties of smart multimedia learning objects, and a set of suitable values for each property. By customizing smart objects, exactly one of these values will be assigned to its related property. An example is the dynamic metadata field “scenario” where the content customizer allows to select a mode (for example guided tour) and a name of an already stored scenario. In order to be able to offer this functionality, we implemented a content customizer tool. Figure 3 shows the graphical user interface of the content customizer. The dynamic metadata fields are grouped into categories in order to offer a good user interface design. While the fields of custom properties can be used for content dependent parameter, all other fields are content independent parameter.

Once a visualization unit (applet) has been retrieved, the content customizer loads its dynamic metadata from a specific database. An interactive visualization can then be parametrized by selecting the necessary values for each parameter. The result can be stored in XML or HTML syntax.



User interface of the content customizer

5. Conclusion

In this paper we described an approach which allows the adaptation of the visualization content according to user needs. In contrast to existing approaches, the educator can adjust both, the level of explanation and the level of interactivity of an animation, thus influencing the presentation and the results of the algorithms being illustrated according to a desired level of functionality and appearance, suitable for the specific needs of the students.

Our approach is based on the middleware approach. In this three-tier model, the lowest layer of this paradigm suits the programmer. The user interface of the current educational visualization remains as the top-layer medium to which the end-user (student) interacts with. On the middle layer (e.g. middleware) acts the educator. This level is composed of various services common to all educational animations, such as management of data, control, interaction level and presentation.

Our experience suggests that the visualizations offered by our systems may well provide an environment in which an educator without conventional programming skills can build a useful interactive visual algorithm relevant to a particular task. The systems will therefore continue to be extended, particularly by increasing the available choice of visualization units. Other activities will be:

- making empirical studies to measure the effectiveness of the V-J Lab in order to improve learning skills in the area of Computer Science;
- implementing usability studies to asses the effectivity of the applets;
- producing standards guides for future user interface designs;
- building a front – end system based on the Web for facilitating and supporting the students use.

References

[1] Abdulmotaleb El Saddik, Stephan Fischer, and Ralf Steinmetz, "ITBeankit: An Educational Middleware Framework for Bridging Software Technology and Education". In Proccedings of EdMedia 2000, Montreal, Canada, June 2000.

[2] Javier Diaz, Claudia Queiruga, Laura Fava, "Trends in Building GUIs in Java: the Approach Used in an Advanced Course of Computer Science", acepted for INC2001

[3] Javier Diaz, Claudia Queiruga, Laura Fava, Claudia Villar, "A Virtual Java-simulation Lab for Computer Science Students", acepted for WebNet2000

[4] Abdulmotaleb El Saddik, Cornelia Seeberg, Achim Steinacker, Klaus Reichenberger, Stephan Fischer, and Ralf Steinmetz. "A Component-based Construction Kit for Algorithmic Visualizations", In Proceedings of the INTEGRATED DESIGN & PROCESS TECHNOLOGY "IDPT'99", June 1999. ISBN 1-880094-35-5.

[5] Díaz J., Queiruga, C.; Harari, I.(1996), *"Testing the Usability of a Hypermedia Enclyclopedia in Data Networks"*, PANEL'96 (XXII Conferencia Latinoamericana de Informática), Universidad de los Andes, Bogotá, Colombia.

[6] Díaz, J.; Queiruga C.; Schiavoni A. (1994), "*A proposed extension to the hypermedia model for training in networking technology"*, BIWIT'94 (Basque International Workshop on Information Technology), Biarritz, Francia.

[7] Multibook, http://www.multibook.de

[8] Gamma E. , Helm R. ,Johnson R. , Vlissides J. : "Design Patterns - Elements of reusable Object-Oriented Software", Addison Wesley ISBN 0-201-63361-2, 1995

[9] Stasko J., Badre A. and Lewis C., "Do Algorithm Animations Assist Learning? An Empirical Study and Analysis", ACM Interchi Conference Proceedings, 1993.

[10] IEEE Learning Technology Standards Committee (LTSC): Learning Object Metadata (LOM): http://www.manta.ieee.org/p1484/

[11] Böhm K. and Rakow T. . "Metadata for Multimedia Documents". In Metadata for Digital Media, Special issue of SIGMOD record, 23 (4), ACM Press, December 1994.